UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/658,684	09/08/2003	Joachim Otto	09700.0074-00	2143

60668 7590 02/06/2008

SAP / FINNEGAN, HENDERSON LLP
901 NEW YORK AVENUE, NW
WASHINGTON, DC 20001-4413

EXAMINER	
INGBERG, TODD D	
ART UNIT	PAPER NUMBER
2193	

MAIL DATE	DELIVERY MODE
02/06/2008	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/658,684	OTTO ET AL.	
	Examiner	Art Unit	
	Todd Ingberg	2193	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *01 November 2007*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-20* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-20* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *6/11/04* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☒ Other: _____.

DETAILED ACTION

Claims 1 – 20 have been examined

Claims 2, 16 and 19 were amended.

Information Disclosure Statement

1. Within the rejection under 103 below the Examiner believes an Information Disclosure Statement on the named products is relevant to the case. Some reference showing the two languages the Applicant is indicating (making a legal claim) makes the invention novel or non-obvious. Documents relevant to the claimed invention on SAP Web Dynpro are requested.

Claim Rejections - 35 USC § 101

2. Prior rejection under 35 U.S.C. 101 has been overcome by amendment for claims.

Claim Rejections - 35 USC § 102

3. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

4. Claims 1 -20 are rejected under 35 U.S.C. 102(b) as anticipated by or, in the alternative, under 35 U.S.C. 103(a) as obvious over by Template Software.

The **Template** product line contains:

The SNAP programming language (One manual used)

The Workflow Template (Not used in this Office Action)

The Web Component (One manual used)

These three layered products work together.

The documentation sets for the products contains the following manuals.

SNAP released June 1997

SNAP Language Reference (Not used in this Office Action)

Using the SNAP Language (Not used in this Office Action)

Using the SNAP Communication Component (Not used in this Office Action)

Using the SNAP Graphic User Interface Component (Not used in this Office Action)

Getting Started with SNAP (Not used in this Office Action)

Using the SNAP Display Editors (Not used in this Office Action)

SNAP Class Library Reference (Not used in this Office Action)

Using the SNAP External Application Software Component (Not used in this Office Action)

Using the SNAP Development Environment (Referred to as **SNAP**)

SNAP Module Library Reference (Not used in this Office Action)

Using the SNAP Permanent Storage Component (Not used in this Office Action)

Workflow released September 1997

Developing a WFT Workflow System (Not used in this Office Action)

Using the WFT Development Environment (Not used in this Office Action)

WFT Library Reference (Not used in this Office Action)

Web Component

Using the Web Component (Referred to as **WEB**)

Training

Foundation Template SNAP 8.0 (**TRAIN**)

Since, these products work together they constitute a single reference and can be used as the basis for a rejection based on anticipated by a product offering.

Claim 1

Template anticipates a computer program product, tangibly embodied in a storage device, the computer program product being operable to cause data processing apparatus to perform operations comprising: receiving an original design-time representation of an application (SNAP, page 4-8, Creating a GUI) , the original design time representation for use in a first run-time environment for executing applications having been developed in a first design-time environment (SNAP, page 4-9, Steps to make a GUI) , the first design-time environment using a first programming model comprising one or more first model elements including screens and processing logic for each screen, the original design-time representation including one or more application screens (SNAP, pages 4-10 to 4-25) and original processing logic for each application screen (SNAP, page 4-13, classes); generating a converted design-time representation of the application based on the original design-time representation (SNAP, page 4-9 and See SNAP pages 2-54 to 2-55) , WEB teaches the converted design-time representation for use in a second run-time environment for executing applications having been developed in a second design-time environment (WEB, Chapter 2) , the second design-time environment using a second programming model comprising one or more second model elements including models (WEB, pages 2-13 to 2-19), views, and controllers (WEB, pages 2-3 to 2-5), the converted design-time representation including one or more application views based on the one or more application screens (WEB, pages 2-3 to 2-5), and converted processing logic based on the original processing logic (WEB, pages 2-3 to 2-5) , the converted processing logic capable of being executed in the second run-time environment; and storing the converted design-time representation of the application in a repository (WEB, pages 2-1 to 2-19).
SNAP teaches a development environment that can design solutions and generate executable code for a first environment as either an executable and/or HTML. Web is a add-on that can convert the output of SNAP and make the solution Web based (second environment). Therefore it would have been obvious to one of ordinary skill in the art at the time of invention to one of ordinary skill to implement solutions for two environments using the Template system because the ability to produce code for local and web based solutions is more flexible.

Claim 2

The computer program product of claim 1, wherein the first programming model is client-server programming model (TRAIN, page 7-7), and the second programming model is Web programming model. (WEB, pages 2-1 to 2-19).

Claim 3

The computer program product of claim 1, wherein generating a converted design-time representation of the application comprises: converting each application screen to a

corresponding application view ; and converting the original processing logic for each application screen to the converted processing logic. As per claim 1.

Claim 4
The computer program product of claim 3, wherein: each application screen comprises one or more controls from a first set of controls defined in the first programming model; the second programming model defines a second set of controls ; and converting each application screen comprises selecting a corresponding control from the second set of controls for each control in the application screen As per claim 1.

Claim 5
The computer program product of claim 4, wherein each control comprises an attribute , and wherein converting each application screen further comprises, for each control in the application screen , setting the attribute of the corresponding control to match the attribute of the control in the application screen As per claim 1.

Claim 6
The computer program product of claim 3, wherein the original processing logic comprises state control logic and one or more calls to one or more run-time modules in the first run-time environment , and wherein converting the original processing logic comprises: generating corresponding state control logic that is executable by an adapter in the second run-time environment , the adapter being operable to interface with the run-time modules in the first run-time environment ; and converting the calls to the run-time modules into instructions to the adapter for invoking the run-time modules As per claim 1.

Claim 7
The computer program product of claim 3, wherein converting the original processing logic comprises generating one or more instructions to an adapter in the second run-time environment to perform a function not performed by the original processing logic As per claim 1.

Claim 8
The computer program product of claim 3, wherein: converting the original processing logic comprises generating code to invoke an adapter in the second run-time environment; and the code to invoke the adapter is formatted to resemble the original processing logic As per claim 1.

Claim 9
The computer program product of claim 8, wherein the code to invoke the adapter comprises one or more macros (As per claim 1 – code to convert).

Claim 10
A system comprising: a first run-time environment operable to execute run-time code generated from design-time representations of applications developed in a first design-time environment, the first design-time environment using a first programming model comprising one or more first

model elements including models, views, and controllers; a conversion module operable to: receive an original design-time representation of an application, the original design-time representation for use in a second run-time environment for executing applications having been developed in a second design-time environment, the second design time environment using a second programming model comprising one or more second model elements including screens and processing logic for each screen, the original design-time representation including one or more application screens and original processing logic for each application screen, the original processing logic including a call to a run-time module in the second run-time environment; and generate a converted design-time representation of the application based on the original design-time representation, the converted design-time representation for use in the first run-time environment, the converted design-time representation including one or more application views based on the one or more application screens, and convert; processing logic based on the original processing logic, the converted processing logic capable of being executed in the first run-time environment; and ate adapter operable to interface with the run-time module in the second run-time environment. As per the rejection for claim 1.

Claim 11
The system of claim 10, wherein the converted processing logic comprises an instruction to the adapter to invoke the run-time module based on the call to the run-time module in the original processing logic. As per the rejection for claim 1.

Claim 12
The system of claim 10, wherein: the first programming model defines a first set of controls; the second programming model defines a second set of controls; and the converted design-time representation of the application comprises a corresponding control from the first set of controls for each control in the original design-time representation of the application. As per the rejection for claim 1.

Claim 13
The system of claim 10, wherein the converted processing logic comprises instructions that are formatted to resemble the original processing logic. As per the rejection for claim 1.

Claim 14
The system of claim 10, wherein the converted design-time representation of the application comprises additional processing logic not included in the original processing logic. As per the rejection for claim 1.

Claim 15
An apparatus comprising: means for receiving an original design-time representation of an application, the original design-time representation for use in a first run-time environment for executing applications having been developed in a first design-time environment, the first design-time environment using a first programming model comprising one or more first model elements including screens and processing logic for each screen, the original design-time representation including one or more application screens and original processing logic for each

application screen; and means for generating a converted design-time representation of the application based on the original design-time representation, the converted design-time representation for use in a second run-time environment for executing applications having been developed in a second design-time environment, the second design-time environment using a second programming model comprising one or more second model elements including models, views, and controllers, the converted design-time representation including one or more application views based on the one or more application screens, and converted processing logic based on the original processing logic, the converted processing logic capable of being executed in the second run-time environment. As per the rejection for claim 1.

Claim 16

The apparatus of claim 15, wherein the first programming model is client-server programming model, and the second programming model is Web programming model. As per the rejection for claim 2.

Claim 17

The apparatus of claim 15, wherein the means for generating a converted design-time representation of the application comprises: means for converting each application screen to a corresponding application view; and means for converting the original processing logic for each application screen to the converted processing logic. As per the rejection for claim 1.

Claim 18

A method comprising: receiving an original design-time representation of an application, the original design time representation for use. in a first run-time environment for executing applications having been developed in a first design-time environment, the first design-time environment using a first programming model comprising one or more first model elements including screens and processing logic for each screen, the original design-time representation including one or more application screens and original processing logic for each application screen; and generating a ~ -averted design-time representation of the application based on the original design-time representation, the converted design-tune representation for use in second run-time environment for executing applications having been developed in a second design-time environment, the second design-time environment using a second programming model comprising one or more second model elements including models, views, and controllers, the converted design-time representation including one or more application views based on the one or more application screens, and converted processing logic based on the original processing logic, the converted processing logic capable of being executed in the second run-time environment. As per the rejection for claim 1.

Claim 19

The method of claim 18, wherein the first programming model is client-server programming model, and the second programming model is Web programming model. As per the rejection for claim 2.

Claim 20

The method of claim 18, wherein generating a converted design-time representation of the application comprises: converting each application screen to a corresponding application view; and converting the original processing logic for each application screen to the converted processing logic. As per the rejection for claim 1.

Response to Arguments

5. The following are the Applicant's remarks with the Examiner's response.

Applicant's Remarks

"I. Preliminary Matter

Applicants thank the Examiner for the courtesy and professionalism extended during the telephonic interview with Applicants' representative on September 19, 2007. As a preliminary matter, Applicants believe that the Office Action mailed August 1,2007 ("Office Action") is incomplete with respect to the§ 102(b) and § 103(a) rejections because the Examiner repeated the rejections without having considered Applicants' arguments• As M.P.E.P. § 707.07(f) makes clear:

In order to provide a complete application file history

and to enhance the clarity of the prosecution history record, an examiner must provide clear explanations of all actions taken by the examiner during prosecution of an application.

Where the applicant traverses any rejection, the

examiner should, if he or she repeats the rejection, take note

of the applicant's argument and answer the substance of it.

Applicants, however, in an effort to advance prosecution, reply to the Office Action with substantially same arguments made in response to the Final Office Action mailed February 23, 2007 ("Final Office Action"), as requested by the Examiner during the telephonic interview."

Examiner's Response

6. In the Interview of September 19, 2007 the Examiner stated he repeated the rejection because the Applicant's statements in the response appeared the reference was lacking. Examiner also, state the next action would be non final, because the Examiner wanted to ensure the Applicant had the complete reference.

The following is from the prior Office action:

"Applicant's arguments filed June 22, 2007 have been received. The Examiner will consider arguments after the Examiner knows the Applicant has had an opportunity to review the references used in the Office Action"

This action is non-final.

Applicant's Remarks

"II. Regarding the Office Action

In the Office Action, the Examiner requested for documents relevant to the claimed invention on SAP Web Dynpro and SAP Dynpro; rejected claims 1, 3-15, 17-18, and 20 under 35 U.S.C. § 102(b) as being unpatentable over SNAP ("Using the SNAP Development Environment," by Template Software) and WEB ("Using the Web Component," by Template Software); and rejected claims 2, 16, and 19 under 35 U.S.C. § 102(b) or 35 U.S.C. § 103(a) as being unpatentable over the commercial product line by Template Software in view of Development Tools.

Applicants have amended claims 2, 16, and 19. Claims 1-20 are currently pending. Based on the foregoing amendments and the following remarks, Applicants respectfully traverse the rejections of the pending claims.

A. Request for Documents

Applicants have amended claims 2, 16, and 19 to remove the terms "SAP Dynpro" and "SAP Web Dynpro." Thus, documents relevant to the claimed invention on SAP Dynpro and SAP Web Dynpro are not required."

Examiner's Response

Documents are no longer requested based on applicant's response.

Applicant's Remarks

"B. Rejections of Claims 1, 3-15, 17-18, and 20 Under 35 U.S.C. § 102(b) The Examiner asserted that the SNAP programming language and the Web Component are "a single offering and constitute a single reference" because "[o]nce Web functionality is present and the Web component is installed the two are inseparable." Final Office Action at 10. The Examiner also asserted that

"[s]ince... these products work together[,] they constitute a single reference." Office Action at 4. Applicants respectfully submit that the Examiner's assertion is not supported by the references. First, nowhere does SNAP disclose anything related to the Web component, and thus a SNAP application or SNAP Development Environment is operable with or without the Web Component. Further, WEB discloses that the Web Component provides a "gateway, which manages the connection between end user web browsers and the requested, running SNAP application." WEB at 2-4. As shown in detail in Figure 2-1 of WEB, the gateway of the Web Component run as a separate entity between a HTTP server and a running SNAP application, or as a plug-in attached to a HTTP server. WEB at 2-4. Thus, the Examiner's assertion that the Web Component and SNAP application are "inseparable" is based on the Examiner's opinion and not

supported by any disclosure made by the references. Accordingly, Applicants respectfully request that the Examiner support the assertion based on the references before shifting the burden to Applicants to provide compelling case law that deals with dependencies of layered products. Moreover, even if SNAP and WEB can be combined to constitute a single reference as asserted by the Examiner, the combination of SNAP and WEB fails to teach or suggest every claim element recited in independent claim 1.

Examiner's Response

First as a mater of burden. It appears the Applicant is stating the Office must provide case law to the Applicant and Applicant can determine it the case law is compelling.

The person of ordinary skill is a hypothetical person who is presumed to be aware of all the pertinent prior art (PHOSITA). Customer Accessories Inc, v Jeffrey-Allan Ind. Inc., 1 USPQ2d 1196 (Fed Cir. 1986).

The hypothetical person skilled in the art is attributed with knowledge of all prior art in the field of the inventor's endeavor and of prior art solutions for a common problem even if outside of that field – and not of all prior art. In re Nilssen 7 USPQ2d 1500 (Fed Cir. 1998).

The argument that the Examiner should provide compelling case law is not compelling.

Second, the Applicant's arguments on technical merit above. The Applicant provides clear support of a web based runtime environment (the second environment). The rejection includes the section of SNAP where HTML can be generated for solutions not using the Web Component. (See SNAP pages 2-54 to 2-55). It is the intent that the ability to design representations of applications (first format) in SNAP and generate code for a first processor (local environment - HTML or exe format). And the ability to produce run-time code for another (Web Component).

Applicant's Remarks

"In particular, the combination of SNAP and WEB fails to teach or suggest "generating a converted design-time representation of the application based on the original design-time representation, the converted design-time representation for use in a second run-time environment for executing applications having been developed in a second design-time environment, the second design-time environment using a second programming model comprising one or more second model elements including models, views, and controllers, the converted design-time representation including one or more application views based on the one or more application screens, and converted processing logic based on the original processing logic, the converted processing logic capable of being executed in the second run-time environment." (emphasis added)."

Examiner's Response

Examiner disagrees. The narrative in the Examiner response just prior is the basic working of the products. Examiner hopes this clarifies the art and the Applicant can distinguish over the Template Software products.

Applicant's Remarks

"WEB discloses that the Web Component "converts:.. SNAP displays at run time to HTML or Java displays." WEB at 2-9. (emphasis added). Thus, the Web Component does not generate a converted design-time representation of SNAP displays. The Examiner asserted that "the format is not a binary O's and l's but HTML with embedded JAVA... stored in [files]... can be directly manipulated. Final Office Action at 11. Applicants respectfully disagree."

Examiner's Response

The ultimate run-time code must be executable machine code. The machine code is not visible in the development environment. The HTML code can be viewed. That is the basic point the Examiner was attempting to make to enable the Applicant to distinguish over the art.

Perhaps the adding of section of SNAP (See SNAP pages 2-54 to 2-55) will assist the Applicant.

Applicant's Remarks

"First, WEB teaches against storing a converted run-time representation of SNAP displays for later manipulation by disclosing that "(t)he SNAP display web pages exist only as long as they are accessed by end users." WEB at 2-4. (emphasis added). Thus, the converted run-time representation of SNAP displays exists only at run-time and is not stored as files for manipulation at design-time. Moreover, even if a converted run-time HTML representation of SNAP displays can be stored as files and manipulated later at design-time, a converted run-time JAVA representation of SNAP displays is binary (O's and l's) and thus may not be manipulated later at design time. WEB fails to disclose that the Web Component enables a developer to manipulate a converted run-time JAVA representation of SNAP displays. Thus, the Web Component generates, at most, a converted run-time representation of SNAP displays, and fails to generate a converted design-time representation of SNAP displays.

Furthermore, the Web Component generates neither a converted design time representation nor a converted run-time representation of processing logic of a SNAP application. WEB discloses that converting SNAP application to run on the web does not involve "generating a converted design-time representation of the application based on the original design-time representation,.., the converted design- time representation including.., converted processing logic based on the original processing logic... " (emphasis added). Instead, the Web Component generates a

converted run-time representation of SNAP displays only and creates a new object of the WEB ACCEPTER class, which allows a running SNAP application to accept connections from a HTTP server via the Web Component gateways. Id. Thus, processing logic of a SNAP application runs "unconverted" and is only exposed to the web through the converted run-time representation of SNAP displays and the WEB ACCEPTER object. WEB at 2-10. In another words, rather than generating a converted design-time representation of processing logic of a SNAP application, the Web Component merely facilitates exposing the processing logic to the web."

Examiner's Response

The Examiner hopes the adding of the section in the SNAP environment and the information on the role of the Web Component above will help the applicant.

Applicant's Remarks

"This argument is further supported by WEB at 2-10 and 2-11, which details the steps involved in running a SNAP application on the web. At steps 3 and 4, the Web Component connects to a running SNAP application and "converts the end-user request into an event or into data that the running SNAP application understands ....

The Web Component classes do this conversion... [and] the

conversion is transparent to the application's SNAP code, as well." WEB at 2-10. Thus, rather than converting processing logic of a SNAP application, the Web Component converts end user requests for a SNAP application. Also, because the conversion of the end-user request is transparent to the application's SNAP code, the code does not need to know whether the request is originally from the web or from a standard SNAP client application.

At step 5, "[t]he SNAP application processes the event or data, and produces output." WEB at 2-11. (emphasis added). The "SNAP application executes with little or no knowledge of its connection to the web... [and] executes as though its end users were communicating with the application process directly [and not using the web]." Id. "While the application generates displays in the same way as a regular SNAP application does, you [i.e., developer] have developed the displays for the application with the web-imposed features and constraints in mind." Thus, again, there are no changes made to processing logic of a SNAP application to run the application on the web. The processing logic remains "unconverted" within a SNAP application whether the application runs on the web or not, and processes the converted end user request without knowing that the request is from the web. The Web Component further puts the responsibility on developers to develop "web-aware" SNAP displays to enable a SNAP application to run on the web.

At step 6, "[t]he Web Component converts the SNAP displays generated by the [SNAP] application at run time into HTML and sends them to the Web Component gateway." WEB at 2-11.

For these additional reasons, converting a SNAP application to run on the web does not involve "generating a converted design-time representation of the application based on the original design-time representation,.., the converted design-time representation including.., converted processing logic based on the original processing logic..."

The Examiner asserted that WEB at 2-3 to 2-5 discloses "converted processing logic." However, WEB at 2-3 to 2-5 discusses generating a run-time representation of SNAP displays only.

The Web Component enables you to develop SNAP applications whose displays are presented to end users as web pages. Through the Web Component software, the SNAP application's displays are converted at run time into web pages and displayed via end users' web browsers. When end users access these web pages through their browsers, they actually interact with the SNAP application. For SNAP applications, end users submit URL requests to connect to those applications in the same way that they request access to statically defined web pages; however, the URLs used to connect to SNAP applications specify the applications to be accessed and other necessary related information.

"The difference between simple, statically defined web pages and SNAP application displays presented as web pages is that SNAP display web pages are dynamically generated by an interactive program ..... The SNAP display web pages exist only as long as they are accessed by end users, while static web pages exist for some significant amount of time between manual updates.

•.. In contrast, displays used in SNAP applications that run on the web are HTML displays or Java applets generated by the Web Component

Characteristics of the displays and applets that run on the web are as follows:

At run time, the Web Component converts most kind of SNAP

application displays to interactive HTML documents, and

converts other kinds of SNAP displays (charts, canvases, and

topologies) to static Java applets ....

• SNAP applications on the web use sophisticated URL addressing schemes to prevent end users from accessing unauthorized portions of SNAP applications or other end users' application data.

• Certain features of SNAP do not operate over the web, because of the structure and limitations of HTML and of the various web browser programs.

WEB at 2-3.

WEB at 2-4 discusses the structure of SNAP applications on the web: the end users' web browsers, a web server, the Web Component gateway, and a running SNAP application. WEB at 2-5 discusses the end user's web browsers, the web server, and the Web Component gateway in more detail. Therefore, nowhere does WEB at 2-3 to 2-5 teach or suggest "generating a converted design-time representation of the application based on the original design-time representation,.., the converted design-time representation including.., converted processing logic based on the original processing logic..."

For at least these reasons, SNAP and WEB, taken alone or in combination, fail to teach or suggest every claim element recited in independent claim 1. Independent claims 10, 15, and 18recite features that are similar to the features recited in independent claim 1. Accordingly, Applicants respectfully request reconsideration and withdrawal of the § 102 rejection of independent claims 1, 10, 15, and 18 based on SNAP and WEB.

Examiner's Response

It appears the Applicant is arguing more than claimed in the independent claims. It is unclear if the Applicant when arguing the differences above:

""The difference between simple, statically defined web pages and SNAP application displays presented as web pages is that SNAP display web pages are dynamically generated by an interactive program ..... The SNAP display web pages exist only as long as they are accessed by

end users, while static web pages exist for some significant amount of time between manual updates." Is meaning to claim collaborative transparency, virtualization of the basic steps the rejection covers. The Examiner does not see the arguments in the claim limitations.

Applicant's Remarks

"Claims 3-9 depend from claim 1, claims 11-14 depend from claim 10, claim 17 depends from claim 15, and claim 20 depends from claim 18. Thus, dependent claims 3-9, 11-14, 17, and 20 are allowable by virtue of their dependence on an allowable independent claim. Accordingly, Applicants respectfully request reconsideration and withdrawal of the § 102 rejection of dependent claims 3-9, 11-14, 17, and 20 based on SNAP and WEB."

Examiner's Response

For the same reasons the independent claims are rejected the dependent claims are also rejected,

Applicant's Remarks

"C. Rejections of Claims 2, 16, and 19 Under 35 U.S.C. §§ 102(b) or 103(a) The Examiner rejected claims 2, 16, and 19 under 35 U.S.C. § 102(b) or 103(a) "as being unpatentable over the commercial product line by Template Software in view of Development Tools." Final Office Action at 8. Claim 2 depends from claim 1, claim 16 depends from claim 15, and claim 19 depends from claim 18. Accordingly, claims 2, 16, and 19 are allowable at least by virtue of their dependence on allowable independent claims 1, 15, and 18, respectively.

With respect to the § 103 rejection, the Examiner asserted that "Template teaches the ability to build GUIs in SNAP to run as a local application and how to Web enable them with the product WEB which enables them to run in another environment." Final Office Action at 8. Applicants respectfully submit that building a local application and "Web enabling" the local application is

not the same as and does not necessarily involve "generating a converted design-time representation of the application based on the original design-time representation..." (emphasis added). As explained above with respect to independent claim 1, WEB teaches web enabling an application by converting an end-user request to an event or data that the application understands and converting output displays generated by the application at run-time to HTML pages without generating a converted design-time representation of the application. Therefore, the Examiner has not established prima facie case of obviousness with respect to claim 2, 16, and 19. Accordingly, Applicants respectfully request reconsideration and withdrawal of the §§ 102 or 103 rejection of claims 2, 16, and 19."

Examiner's Response

For the same reasons the independent claims are rejected the dependent claims are also rejected,

Disposition

The Examiner ensured the Applicant received the references and apologizes if references were missing. Because the large size of the reference and indication some part was missing the Examiner had repeated the rejection. And the Examiner out of fairness told the Applicant's Representative this action was planned to be non final. Applicant has pointed out the requirement the action be non-final which the Examiner was aware of. In an effort to advance the prosecution over the Template product line. the Examiner being a former employee of the company has provided an affidavit, Examiner hopes this will put to rest the dependency issue.

Correspondence Information

7. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Todd Ingberg whose telephone number is (571) 272-3723. The examiner can normally be reached on during the work week..

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Meng-Ai An can be reached on (571) 272-3756. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

Todd Ingberg
Primary Examiner
Art Unit 2193

TI

Affidavit

Commissioner of Patents and Trademarks
Record of Patent Prosecution 10/658,684

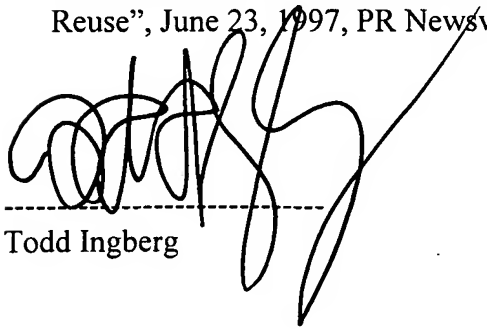
For the Record;

I. Todd Ingberg, declare as follows:

1. I possess first hand knowledge of the Template Software product line. Specifically and only, Version 8.0 of the Template Software product line. The components of Version 8.0, I have knowledge of are the following three components: SNAP, Workflow (referred to as WFT) and Web Component. I have not installed any other products mentioned in Exhibit A.

2. I was employed by and trained by Template Software Inc. during the years 1997 and 1998. I installed the Version 8.0 components mentioned above. The components have a dependency for installation. The SNAP component (a programming environment) must be installed in order to install the Workflow product. The Web Component product is not standalone.

3. The release of the three components of the product line (mentioned above) reflect the ordered dependency. Exhibit A is a Press Release, showing the release date for the components. The SNAP component is the first component released the Workflow Component and the Web Components are released later. Exhibit A is the Press Release listed on the PTO-892 titled, "Template Software Strengths Core Product Family With Ease-Of-Use and Functional Enhancements That Promote Unparalleled Software Reuse", June 23, 1997, PR Newswire.



Todd Ingberg